CS 1050 Homework 2

Spring 2017

Due: 17th March 2017 5:00 PM (No Extension)

Purpose

1) Call and return functions in and out of main.
2) Perform basic array operations. (Creation, indexing, shuffling)
3) Everything else we have done so far.

Submission

Filename must be: labsection-hw2.c (Include your respective lab section)

Example: If you are in lab D,

```
d-hw2.c
```

Submission Command: (Case Sensitive)

```
submit CS1050 HW2 d-hw2.c
```

Note

1) Read the COMPLETE homework document CAREFULLY before starting the assignment.
2) Use function prototypes EXACTLY as given in the document. DO NOT CHANGE ANY FUNCTION PROTOTYPES!
3) Use the sample output to see how to collect information from the user, display menu/options and perform error checking.
4) If anything is not clear in the lab document or if you have questions, ask the TAs for assistance during office hours.

Description

Build a casino where the user can play the following games.

1) Lucky Seven
2) High Card
3) Match Card (Bonus)

The user starts with $100. The user is then prompted to pick a game. All inputs must be error checked. The user is then asked how much they want to bet on the game. (Cannot be 0 or more than the user currently has.) Depending on the user's outcome, the user either wins or loses the amount they bet. All the games require a deck of cards (simulated with an array). The cards will need to be shuffled before each game. The user is kicked out of the casino if they have no money left.

Game Descriptions

Lucky Seven

The user draws two cards. If the sum of the two cards is 7, the user wins 7X the amount bet. If the sum is not 7, the user loses the amount bet.

High Card

The user draws a card from the deck, and the computer picks a random (different) card. If the computer's card is higher, then the player loses. If the player's card is higher, then the player wins. For this version, all cards are different numbers.

Match Card (Bonus Only)

The user does not bet for this game. Works similar to high card, except there will be 4 "decks of cards" in play. (Use a 2D array) The user will choose a deck and a card. The computer will pick a random card from a random deck. If the value of the user's card matches the computer's card, then the user doubles their total money. If the user loses, then the user loses everything.

Random Number Generator

Use the `rand()` function from `<stdlib.h>` to generate random numbers. The output will need to be between 0-12 for card operations. Use `rand()` and the `%` operator. SIZE is a global constant. Treat it like a variable that you can't change. Use it for all card operations where you need the value for size of the deck.

To use the functions srand and rand:

```
#include <stdlib.h>

#include <time.h>

#define SIZE 13
```

Goes at the top of your code, and:

```
srand(time(NULL));
```

As the first line of code inside of `main()`

Win/loss calculation table

Example

For Lucky Seven, if the user has an initial amount of $100, bets $25 and wins, then total = 100 + 7 * 25->$275

Games	Win	Lose
Lucky 7	+ 7 * AmountBet	- AmountBet
High Card	+ AmountBet	- AmountBet
Match Card (Bonus)	+ totalMoney	-totalMoney

If they lose, then the total = 100 - 25 -> $75

Function Prototypes & Descriptions

```
void display_menu();

/*

   Displays the menu for choosing a game. Should only have printf statements.

   Arguments – None

   Return Nothing

*/


int error_check(int);

/*

   Argument – int: option chosen in main

  Check if the options chosen is valid or not

   Returns 1 if input is valid, 0 otherwise.

*/


void shuffle(int[]);

/* Argument – int Array signifying deck of cards, which was created in main

   Return Nothing

   Takes in the deck of cards array, loads with values from 1-13 and shuffles the order. Do not create a new
array. Hint: Pass by reference!

Remember: Index of an array starts from location 0, your deck must have card values from 1-13. To shuffle the
order use rand() function to get a random index and then swap the values. Hint: Bubble sort(but you are not
sorting anything here).

*/


int draw_card(int[]);

/*

   Picks the first card that hasn't been drawn already, marks it as "taken" and

   returns the value of the card.

   Change value to -1 at the location to indicate "taken"

   Arguments— int[] deck of cards, which was created in main()

   return – int: value of card drawn.

*/
```

```
int lucky_seven(int, int);
/*
   Implements lucky seven game logic. Takes in one card value and checks if it is equal to 7.
   return 1 if value matches indicating the user won, return 0 if user loses.
*/
int high_card(int, int);
/*
   High Card game logic. Takes in card values and checks which one is higher.
   return 1 if user card is greater than dealer card indicating user wins, return 0 if user loses.
*/

double calculate_winnings(int, int, double);
/*
   Calculates winnings based on game played and if the user won.
   Arguments- int: game played, int: win or loss, double: amount bet
   return double: amount won or lost.
   Refer the table above.
*/

int match_card(int, int); // Bonus
/*
   Game is only played when the user decides to quit. (Give them a last minute chance to double their money)
   Takes in card values and checks if they are the same.
   return 1 if user wins, return 0 if user loses
*/

int main(void) {}
/*
   Acts as user interaction, creates arrays, calls other functions, keeps track of user's total money.
         First display the menu of games to play.
         Declare all the variable you need to use and also the array representing the deck
```

Every time a game is played, display the amount the user has left and ask the user to enter the amount he wants to bet. Assume the user won't mess up (brownie points if you check this too).

Next, display the result of the game (display if the user won or lost, display the card(s) drawn, and the amount of money gained/lost).

Continue to play the game as long as user does not choose to exit and still has money.

Display final amount of money when the user quits or gets thrown out.

For the bonus, create a 2D Array representing 4 decks of 13 cards each. Shuffle each deck and choose two random cards from two different decks. Display cards drawn and if the user won or lost.

*/

Sample Output

Characters in BOLD are inputs from the user.

```
holt:HW2$ ./a.out


Welcome to the Guilliams Casino!


Which game do you want to play?
1) Lucky Seven
2) High Card
3) Exit
4


Invalid choice; Try again.


Which game do you want to play?
1) Lucky Seven
2) High Card
3) Exit
-1


Invalid choice; Try again.
```

```
Which game do you want to play?
1) Lucky Seven
2) High Card
3) Exit
1
How much do you want to bet? $5
Lucky Seven
You drew 1.
You lost $5.00. Too bad.

Your total money is $95.00.


Which game do you want to play?
1) Lucky Seven
2) High Card
3) Exit
1
How much do you want to bet? $7
Lucky Seven
You drew 1.
You lost $7.00. Too bad.

Your total money is $88.00.

Which game do you want to play?
1) Lucky Seven
2) High Card
3) Exit
2
How much do you want to bet? $50
```

```
High Card
You drew 1. The dealer drew 6.
You lost $50.00. Too bad.

Your total money is $38.00.

Which game do you want to play?
1) Lucky Seven
2) High Card
3) Exit
3

BONUS
Wait! Before you go, do you want a chance to double your money???
1) Yes
2) No
1

Match Card
You drew 3. The dealer drew 11.

You lost everything!!!
Your total is now $0.00.
Thanks for playing!

dpstm3:~$ ./a.out

Welcome to the Guilliams Casino!

Which game do you want to play?
1) Lucky Seven
2) High Card
```

```
3) Exit
3                                                                     // SHOULD EXIT
BONUS
Wait! Before you go, do you want a chance to double your money???    //BONUS ONLY
1) Yes
2) No
2
Bye!
dpstm3:~$ ./a.out

Welcome to the Guilliams Casino!

Which game do you want to play?
1) Lucky Seven
2) High Card
3) Exit
1
How much do you want to bet? $100
Lucky Seven
You drew 2.
You lost $100.00. Too bad.

Your total money is $0.00.

You have been thrown out of the casino.
Be more careful with your money next time.

holt:HW2$ ./a.out
Welcome to the Guilliams Casino!

Which game do you want to play?
```

```
1) Lucky Seven
2) High Card
3) Exit
1
How much do you want to bet? $57
Lucky Seven
You drew 1.
You lost $57.00. Too bad.

Your total money is $43.00.
Which game do you want to play?
1) Lucky Seven
2) High Card
3) Exit
1
How much do you want to bet? $43

Lucky Seven
You drew 1.
You lost $43.00. Too bad.

Your total money is $0.00.
You have been thrown out of the casino.
Be more careful with your money next time!
```

Guidelines for Grading HWK2

60 Points Possible (+5 bonus points)

***** FAILURE TO RETURN THIS DOCUMENT TO THE TA WILL RESULT IN A ZERO GRADE FOR THE LAB *****

General

If your program does not compile, or produce any input/output (I/O) because most of the source code is commented out then your lab will receive a grade of NO POINTS. For any partial credit for labs your C program must not only compile but also produce some valid I/O that meets the lab specifications.

Example: gcc hw2.c –Wall -Werror

0 points – If there is a compile error with –Wall and –Werror

-5 points – Code not properly formatted, Output not formatted correctly, no comments and header at the top (with name, TA and pawprint)
8 points – display menu and error check functions.
10 points – Implementing the shuffle function
10 points – Implementing the draw_card function
10 points – Implementing the calculate_winnings
12 points – Lucky Seven and high_card functions.
10 points – Program follows the sample output logic, loops the game selection accordingly, shows total amount and exits gracefully at every situation.